Санкт-Петербургский Государственный Университет
Фундаментальные информатика и информационные технологии

Профиль математического и программного обеспечения вычислительных машин, комплексов и компьютерных сетей

Солдатов Дмитрий Владимирович

Метод одновременной навигации и составления карты по видео изображению (SLAM) для автономных роботов на ТРИК

Магистерская диссертация

Научный руководитель:
к. ф.-м. н., доц. Вахитов А. Т.

Рецензент:
Кривоконь Д. С.

Санкт-Петербург
2016

SAINT-PETERSBURG STATE UNIVERSITY
Fundamental Computer Science and Information Technologies

Software of Computers, Complexes and Networks

Soldatov Dmitry

Mono visual simultaneous localisation and mapping for autonomous robots on TRIK

Graduation Thesis

Scientific supervisor:
Alexander Vakhitov, Ph.D.

Reviewer:
Dmitry Krivokon

Saint-Petersburg
2016

Оглавление

Введение

Задача одновременной локализации и построения карты (SLAM) для однокамерного зрения может считаться решенной в случае, когда наблюдается неподвижная сцена. Современные методы игнорируют наблюдаемые динамические объекты, хотя в ряде приложений (например, в навигации и управлении мобильными роботами) реконструкция их траектории представляет интерес. В этой работе описываются новые алгоритмы для реконструкции траекторий динамических объектов, а также сравниваются алгоритмы детекции точек динамических объектов. Новые алгоритмы используются на основе траекторий движения камеры, полученных с помощью наиболее точной и современной однокамерной SLAM-системы ORB-SLAM. В работе также предлагается собственный набор данных для тестирования подобных алгоритмов, созданный с помощью контроллера ТРИК, содержащий один динамический объект, движущийся на фоне статической сцены, имеющий наряду с цветными кадрами карты глубин для определения точности работы алгоритмов реконструкции траекторий динамических объектов.

В последние годы появился ряд однокамерных систем одновременной локализации и построения карты (SLAM), работающих в реальном времени, работоспособных на значительном числе видеозаписей из открытых наборов данных разной степени сложности, снятых с помощью ручных и установленных на мобильные роботы камер, при этом получены высокие результаты по точности построения карты и воссоздания траектории движения камеры.

На практике, однако, зачастую используются системы на основе стереокамер или RGB-D сенсоров. Важную роль в объяснении этого факта может играть отсутствие возможностей в современных однокамерных системах SLAM для реконструкции траекторий динамических объектов. Существуют отдельные опубликованные алгоритмы такого рода, однако в основном они либо имеют своей целью реконструировать движение отдельной динамической точки, не обеспечивая при этом точности, сравнимой с точностью для статических точек сцены, либо яв-

ляются теоретическими примерами, которые не были интегрированы в системы SLAM.

В этой работе делается попытка построить на основе современного решения для однокамерного SLAM [9] прототип системы, в которой отлеживаются движущиеся объекты и реконструируются их траектории.

1. Предметная область

В литературе можно найти множество различных подходов к задаче SLAM с использованием разных типов сенсоров: лидары, сонары, камеры [6], в данной работе речь идет об однокамерном SLAM. Однокамерный SLAM - область компьютерного зрения, применяемая в робототехнике для автономной навигации и составления карты местности по видео изображению с одной камеры (в отличии от стерео SLAM, где используется две камеры). Один из подходов однокамерного SLAM заключается в том, что из изображения извлекаются так называемые особые точки (feature points), положение которых затем отслеживаются на последующих изображениях - таким образом формируются треки проекций особых точек. Имея данные о проекциях особой точки на изображения и данные о положении камеры, в момент взятия этих изображений, можно решить задачу вычисления пространственных координат материальной точки, которую можно трактовать как известную в более широком смысле задачу bundle adjustment [4].

1.1. ORB SLAM

Непосредственная реализация однокамероного SLAM - ORB SLAM [9], в которую предполагается дальнейшая интеграция алгоритма реконструирования траекторий динамических объектов, описанного в данной работе, в качестве детектора особых точек использует ORB [1] (Oriented FAST and Rotated BRIEF), который в совокупности с техникой распознавания DBoW2 [5] позволяет обеспечить работу алгоритма в режиме реального времени.

1.2. Отслеживание динамических объектов

Задача детекции и отслеживания динамических объектов (DATMO, также используется термин SLAMMOT) является отдельным направлением исследований [3, 13, 8]. Существуют два подхода к детекции динамических объектов в случае однокамерного зрения, первый из ко-

торых рассматривает индивидуальные траектории точек, в то время как второй заключается в кластеризации полного набора траекторий на группы в соответствии с принадлежностью точек жестким независимо движущимся объектам. Примерами первого подхода являются алгоритмы [13, 8], которые основаны на идее проверки траекторий на соответствие моделям, справедливым для статических точек. Второй подход основан на том факте, что пяти точек на изображении объекта на двух последовательных кадрах достаточно для реконструкции положения камер и реконструкции точек объекта [7]. Эта идея развивается в [11, 14].

Реконструкция положения динамических объектов в рамках современных систем однокамерного SLAM напрямую не выполняется. Существуют алгоритмы реконструкции динамических точек [13, 8], однако не опубликованы сведения о точности этих алгоритмов.

2. Постановка задачи

Дана последовательность изображений I_s, снятых движущейся камерой с известными внутренними параметрами K в моменты времени $t_s, s = 1, 2..N$. Даны евклидовы преобразования R_s, t_s, описывающие положение камер для I_s. Пусть x_i - множество особых точек, наблюдаемых на I_s. Для каждой x_i даны $x_{i,j} \in R^2$ - упорядоченное множество проекций особой точки x_i на I_j, $j \in t_s$. Пусть, $x_{di} \in x_i$ - множество особых точек, принадлежащих динамическим объектам.

1. Разработать алгоритм детекции точек динамических объектов, определяющий множество x_{di} наилучшим образом;

2. Разработать алгоритм реконструкции положения динамического объекта, вычисляющий пространственные координаты $X_{di,j}, j \in t_s$ особых точек динамических объектов x_{di}.

3. Описание алгоритма

Будем считать камеру калиброванной, так что матрица ее внутренних параметров $\mathtt{K}$ известна. Каждая точка x_i порождает трек проекций $\{x_{i,j}\}$ в моменты $j = t_{i,1}, \ldots, t_{i,n_i}, x_{i,j} \in R^2$.

Положение камеры задается матрицей поворота R $3 \times 3, r_{i,j} \in R$ и вектором переноса $\mathbf{t} \in R^3$. Определим вектор параметров θ, как вектор по которому могут быть построены $\mathtt{R}, \mathbf{t}$. Отображение проектирования $\pi(\theta, \mathbf{X})$ для точки $\mathbf{X} \in R^3$ на камеру с положением θ при условии единичной матрицы $\mathtt{K}$ записывается как

$$\pi(\theta, \mathbf{X}) = \frac{1}{\mathbf{r}_3^T \mathbf{X} + t_3} \begin{pmatrix} \mathbf{r}_1^T \mathbf{X} + t_1 \\ \mathbf{r}_2^T \mathbf{X} + t_2 \end{pmatrix}, \tag{1}$$

где $\mathbf{r}_i^T$ - строка i матрицы $\mathtt{R}$, t_i - компонента i вектора переноса, для $i = 1, 2, 3$. Если матрица $\mathtt{K}$ не является единичной, мы можем домножить все проекции в однородном представлении на $\mathtt{K}^{-1}$.

3.1. Определение динамических точек

Если некоторая наблюдаемая камерой точка x_i неподвижна, то ее проекции $x_{i,j} \in R^2$ удовлетворяют уравнению

$$\pi(\theta_{c,j}, \mathbf{X}) = \mathbf{x}_{i,j} + \mathbf{v}_{i,j}, \tag{2}$$

где $\mathbf{v}_{i,j}$ - случайный шум в наблюдениях, $\{\theta_{c,j}\}$ - положения камеры в моменты времени $j = t_{i,1}, \ldots, t_{i,n_i}$.

Точка классифицируется как статическая, если выполнено условие (3) на среднюю ошибку перепроектирования:

$$\exists \mathbf{X_i} \in R^3 : \quad \frac{1}{n_i} \|\pi(\theta_j, \mathbf{X_i}) - \mathbf{x}_{i,j}\| < \epsilon, \tag{3}$$

для некоторого параметра ϵ.

3.2. Реконструкция положения динамического объекта

Динамический объект i состоит из набора точек, представленного в некоторой системе координат объекта $\{\mathbf{X}_{i,j}\}$, $\mathbf{X}_{i,j} \in R^3$. Эти точки движутся согласованно, и их положение в момент t задается вектором параметров $\theta_{o,t}$. Пусть $\cdot$ - операция композиции евклидовых преобразований, тогда $\theta_{oc,j} = \theta_{c,j} \cdot \theta_{o,j}$ задает преобразование из координат объекта в координаты камеры, и проекции каждой точки динамического объекта удовлетворяет уравнению

$$\pi(\theta_{o,t}, \mathbf{X}_{i,k,l}) = \mathbf{x}_{i,kl} + \mathbf{v}_{i,k,l} \; \forall k = 1 \dots N_i, \tag{4}$$

где i - номер объекта, N_i - число точек объекта, l - момент времени.

Начиная с $N_i \geq 5$, по двум кадрам могут быть восстановлены положение и координаты точек объекта. Для $N_i \leq 3$ это невозможно без дополнительных предположений о положении или движении объекта (например, положение или движение в плоскости).

В этой работе будем считать $N_i \geq 5$. Предлагаем следующий алгоритм реконструкции:

1. Если объект наблюдается на двух достаточно удаленных кадрах, рассчитать существенную матрицу по алгоритму [10] для точек объекта, определить консенсусное множество точек, движущихся согласованно с объектом, и отбросить выбросы, инициализировать координаты точек объекта

2. Для последующих кадров, запустить метод OPnP [12] для точек объекта и определить его позицию на каждом кадре

3. Выполнить bundle adjustment для уточнения положения объекта и точек на каждом кадре

4. Особенности реализации

4.1. Прототипирование алгоритма

Алгоритм прототипировался на языке C++, с использованием библиотек OpenCV 3.1, boost 1.58, Ceres Solver [2]. В качестве входных данных программа принимает последовательность изображений тестового видео и список положений камеры, описываемых $\mathbf{R}, \mathbf{t}$ для каждого кадра, полученный из ORB SLAM.

На момент реализации прототипа было принято решение временно отказаться от обработки видео потока в режиме реального времени, поскольку главной целью ставилась проверка работоспособности и оценка точности алгоритма.

Поскольку алгоритм не сразу интегрировался в ORB SLAM, возникла необходимость реализовать собственный трекер особых точек. В качестве детектора и трекера особых точек были взяты FAST и пирамидальный алгоритм построения оптического потока Лукаса-Канаде - их связка проста в реализации и обеспечивает достаточную для исследований точность. В совокупности, FAST и оптический поток дают менее точные результаты, чем аналогичная часть в ORB SLAM. Например, никак не обрабатывается тот случай, когда точка на время покидает поле зрения (выходит за рамки изображения или перекрывается другими объектами) и вновь появляется в кадре. ORB-дескриптор позволяет распознать точку, как ранее наблюдаемую, в то время как оптический поток будет считать ее новой точкой. Однако, к качеству исследуемых в данной работе алгоритмов это не имеет отношения.

В начале работы программы список наблюдаемых особых точек пуст, поэтому на первом изображении последовательности безусловно выполняется детектирование особых точек. В дальнейшем изображения разбиваются на квадранты и в случае недостатка точек в квадранте они доопределяются в нем. Количество отслеживаемых точек со временем может уменьшаться по причине того, что точка вышла из поля зрения или мера ее качества (определяемая функцией calcOpticalFlowPyrLK())

стала слишком маленькой.

С помощью пирамидального алгоритма Лукаса-Канаде находится смещение особых точек - таким образом формируется трек проекций особых точек - упорядоченное множество проекций особой точки отслеживаемой от кадра к кадру.

4.1.1. Реализация алгоритма определения динамических точек

Имея достаточно длинный трек проекций особой точки, можно выполнить алгоритм детекции точек динамических объектов, теория которого описана в 3.1. Полагаем, что трек достаточно длинный, если угол между главными осями камер крайних изображений больше некого значения. Практически, алгоритм реализован следующим образом: вычисляется приближенное положение точки в пространстве с помощью функции opencv TriangulatePoints() (в которой, в частности, используются данные о положении камеры, полученные из ORB SLAM), затем координаты точки уточняются посредством решения задачи bundle adjustment [4] с помощью ceres solver. В конечном итоге, вычисляется ошибка обратного проектирования точки, для которой было выбрано три меры:

1. средняя ошибка обратного проектирования на первый и последний кадр траектории точки (средняя по двум);

2. средняя ошибка обратного проектирования на три кадра (средняя по трем);

3. наибольшая ошибка обратного проектирования.

На основе этой ошибки точка определяется как кстатическая или динамическая. Ниже представлены результаты работы классификатора динамических точек: [Результаты с ROC-кривыми сюда?]

4.1.2. Реализация алгоритма реконструкции положения динамического объекта

Имея треки проекций динамических точек, можно выполнить алгоритм реконструкции траектории динамического объекта (3.2). Проекции динамических точек на двух достаточно удаленных друг от друга изображениях подаются функции opencv findEssentialMat(), которая возвращает существенную матрицу, из которой в свою очередь можно получить евклидово преобразование координат первого кадра в координаты второго с помощью функции recoveryPose(). Стоит заметить, что в случае наблюдения нескольких динамических объектов для определения преобразования будут взяты точки только одного из объектов в силу того, что findEssentialMat() использует алгоритм RANSAC для формирования консенсусного множества. Имея вышеописанное преобразование, можно найти пространственные координаты точек объекта относительно первого кадра. Аналогично случаю, описанному в предыдущем разделе, только уже для группы точек, выполняется функция triangulatePoints() и затем формируется задача bundle adjustment для ceres solver. Далее для выбранной группы точек динамического объекта и каждого изображения можно решить задачу PnP (например, [12]), которая по пространственным координатам точек и их проекциям на изображение дает евклидово преобразование, описывающее преобразование координат объекта в координаты камеры. Как было описано в 3.2 можно получить положение объекта в мировых координатах, составив композицию соответствующий преобразований. Выполняя эту процедуру для каждого кадра можно получить траекторию группы точек динамического объекта, которую уточняется формированием еще одной задачи bundle adjustment для ceres solver.

4.2. Google Ceres Solver

Ceres Solver - это открытая библиотека, написанная на C++, предназначенная для моделирования и решения больших, сложных задач оптимизаций (в частности, bundle adjustment). Это многофункциональ-

ная, развитая и производительная библиотека. В рамках исследуемых алгоритмов с помощью этой библиотеки решались следующие задачи:

1. Уточнение пространственных координат точки в алгоритме детекции точек динамических объектов. Уточняемые значения: координаты точки $\mathbf{X} \in R^3$. Известные значения: N проекций особой точки $x_i \in R^2$; N положений камеры, описываемых вектором длиной 6: 3 - вектор вращения $\mathbf{r}(\mathbf{R})$, 3 - вектор смещения $\mathbf{t}$;

2. Уточнение пространственных координат группы точек динамического объекта в алгоритме реконструкции положения динамического объекта. Уточняемые значения: $M >= 5$ координат точек $\mathbf{X}_i \in R^3$. Известные значения: $N \times M$ проекций особых точек $\mathbf{x}_{i,j} \in R^2$; N положений камеры, описываемых вектором длиной 6: 3 - вектор вращения $\mathbf{r}$, 3 - вектор смещения $\mathbf{t}$;

3. Уточнение траекторий группы точек в алгоритме реконструкции положения динамического объекта. Уточняемые значения: M >= 5 координат точек $\mathbf{X}_i \in R^3$; N положений камеры, описываемых вектором длиной 6: 3 - вектор вращения $\mathbf{r}$, 3 - вектор смещения $\mathbf{t}$. Известные значения: NxM проекций особых точек $\mathbf{x}_{i,j} \in R^2$.

4.3. Калибровка камеры

Для получения внутренних параметров камеры - фокус, оптический центр, коэффициенты дисторсии - выполнялась калибровка камеры с помощью калибровочной доски и Camera Calibration Toolbox for Matlab[1].

4.4. Docker

ORB SLAM требует наличие ROS Indigo - промежуточное ПО, ориентированное на разработку приложений для роботов, требующее в

[1] www.vision.caltech.edu/bouguetj/calib_doc/

свою очередь Ubuntu 14.04. Поэтому было принято решение взять официальный docker-образ osrf/ros:indigo-desktop-ful с доставлением пакетов BLAS, LAPACK, Eigen3, boost, opencv, suitsparse. Docker[2] - легковесная альтернатива виртуальным машинам на базе систем виратуализации на уровне операционной системы, таких как LXC [3]. ORB SLAM имеет графический интерфейс в виде окон отображения особых точек на текущем видео и среды визуализации трех-мерной карты ros rviz. Поэтому docker-машине был предоставлен доступ к X server с помощью утилиты xhsot:

```
>xhost +local:ros-orb-slam
>docker run -it -h ros-orb-slam  \
  --env=''DISPLAY'' \
  --volume=''/tmp/.X11-unix:/tmp/.X11-unix:rw'' \
  ros-orb-slam
```

, где ros-orb-slam - docker-образ.

4.5. Набор данных

Для тестирования алгоритма были записаны тестовые видео с помощью сенсора Kinect разрешением 640×320, снабженные картами глубин для каждого кадра, на которых наблюдается некая статическая сцена, пригодная для выделения особых точек, и один или более недеформируемых динамических объектов - роботизированные тележки, движущиеся прямолинейно. Набор данных содержит:

- 3 видео записи с картами глубин, наблюдающих один динамический объект, предназначенные для определения точности реконструкции траекторий динамических объектов: 2 записи с наездом камеры на динамический объект, 1 запись с параллельным движением;

[2] www.docker.com

[3] https://en.wikipedia.org/wiki/LXC

Рис. 1: Примеры изображений набора данных

- 2 видео записи с картами глубин, наблюдающих два динамических объекта, движущихся в разных направлениях, предназначенные для определения точности формирования консенсусного множества точек множества динамических объектов.

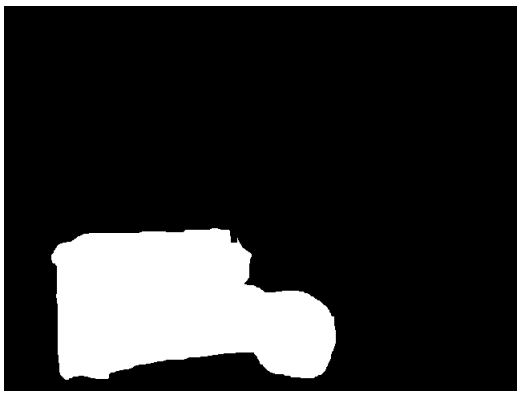
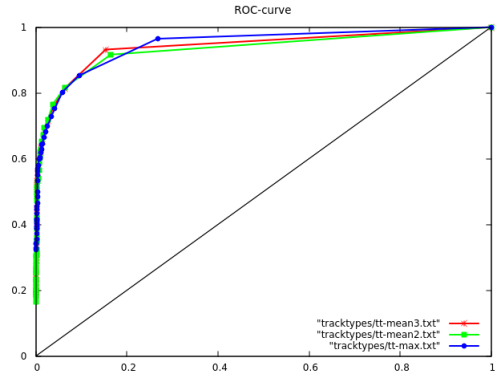
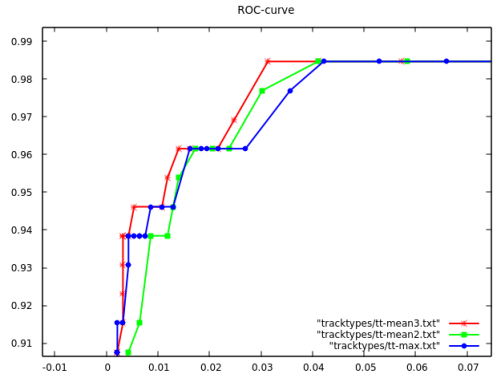


Рис. 2: Маска динамического объекта.



(a) для каждого 20ого кадра.



(b) для последнего кадра (увеличено).

Рис. 3: ROC-кривые, где TP - истинно динамические точки, TN - истинно статические точки. Классификация по наибольшей ошибке показала незначительно лучший результат.

5. Результаты экспериментов

Качество алгоритма определения динамических точек определялось посредством ручного формирования маски динамического объекта. Ниже представлены метрики качества для трех типов классификатора, описанных в . Были проведены эксперименты с указанием маски динамического объекта для последнего кадра записи и указанием маски для каждого 20ого кадра записи (в целях приблизительной оценки работы алгоритма в режиме реального времени)

Большую негативную часть в качество классификатора вносят так называемые ”плавающие точки”, являющиеся особенностью работы оптического потока, когда изначально статические точки ”перескакивают” на динамический объект. Так же классификатор плохо показал себя на

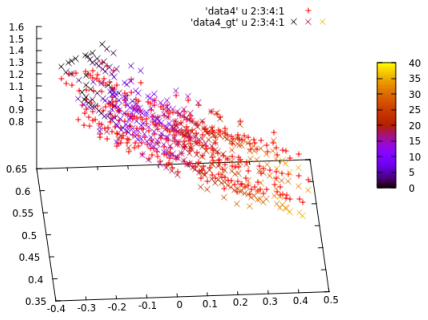


Рис. 4: Совмещенные графики пространственных координат полученных в результате алгоритма и с помощью сенсора Kinect

записи, где камера движется параллельно динамическому объекту.

Точность реконструкции траектории динамического объекта определялась сравнением полученных алгоритмом пространственных координат динамических точек объекта с координатами, восстановленными из карты глубин, полученной с помощью сенсора Kinect.

[График покадровой разности расстояния до точек динамического объекта]

6. Заключение

В результате проделанной работы были выполнены следующие задачи:

- Разработан алгоритм и реализован прототип детекции точек динамических объектов на базе ORB SLAM. Было предложено несколько способов классификации точек, были проведены эксперименты, в которых исследовалось качество каждого из предложенных способов;
- Разработан алгоритм и реализован прототип реконструкции траекторий точек динамических объектов на базе ORB SLAM. Были проведены эксперименты, в которых исследовалось качество алгоритма в сравнении с данными из карт глубин.

Список литературы

[1]

[2] Agarwal Sameer, Mierle Keir, Others. Ceres Solver. — http://ceres-solver.org.

[3] Azim Asma, Aycard Olivier. Detection, classification and tracking of moving objects in a 3D environment // Intelligent Vehicles Symposium (IV), 2012 IEEE / IEEE. — 2012. — P. 802–807.

[4] Bill Triggs Philip McLauchlan Richard Hartley Andrew Fitzgibbon. Bundle Adjustment — A Modern Synthesis. — 2006. — URL: https://lear.inrialpes.fr/pubs/2000/TMHF00/Triggs-va99.pdf.

[5] Dorian Gálvez-López Juan D. Tardós. Bags of Binary Words for Fast Place Recognition in Image Sequences // IEEE Transactions on Robotics, vol. 28. — 2012.

[6] Durrant-Whyte H., Bailey T. Simultaneous localization and mapping, part i // Robotics and Automation Magazine, IEEE, vol. 13. — 2006.

[7] Hartley R., Zisserman A. Multiple view geometry in computer vision. — Cambridge university press, 2003.

[8] Hsiao Chen-Han, Wang Chieh-Chih. Achieving undelayed initialization in monocular slam with generalized objects using velocity estimate-based classification // Robotics and Automation (ICRA), 2011 IEEE International Conference on / IEEE. — 2011. — P. 4060–4066.

[9] Mur-Artal Raúl, Tardós Juan D. ORB-SLAM: Tracking and mapping recognizable features // MVIGRO Workshop at Robotics Science and Systems (RSS), Berkeley, USA. — 2014.

[10] Nistér David. An efficient solution to the five-point relative pose problem // Pattern Analysis and Machine Intelligence, IEEE Transactions on. — 2004. — Vol. 26, no. 6. — P. 756–770.

[11] Reconstructing 3D independent motions using non-accidentalness / Kemal Egemen Ozden, Kurt Cornelis, Luc Van Eycken, Luc Van Gool // Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on / IEEE. — Vol. 1. — 2004. — P. I–819.

[12] Revisiting the PnP problem: A fast, general and optimal solution / Y. Zheng, Y. Kuang, S. Sugimoto et al. // Computer Vision (ICCV), 2013 IEEE International Conference on / IEEE. — 2013. — P. 2344–2351.

[13] Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments / Davide Migliore, Roberto Rigamonti, Daniele Marzorati et al. // Proceedings of International workshop on Safe navigation in open and dynamic environments application to autonomous vehicles. — 2009.

[14] Vidal René, Sastry Shankar. Optimal segmentation of dynamic scenes from two perspective views // Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on / IEEE. — Vol. 2. — 2003. — P. II–281.